Divide and Conquer

$$\underbrace{7 \quad 8}_{DC} \;\Big|\; \underbrace{10 \quad 1 \quad 4}_{DC}$$

7 8 1 4 10

1 4 7 8 10

Goal $\Theta(n \log(n))$

Merge Sort

14 7 3 12 | 9 11 6 2

ms ms

O(1)

14 7 3 12 9 11 6 2

MS MS MS MS

O(1)

14 7 3 12 9 11 6 2

merge merge merge merge

O(n)

7 14 3 12 9 11 2 6

merge merge

O(n)

3 7 12 14 2 6 9 11

merge

O(n)

2 3 6 7 9 11 12 14

$\log_2(8) = 3$

$O(n \log(n))$

$n/2 = 5/2 = 2$

$\underbrace{5 \quad 2}_{ms} \;\Big|\; \underbrace{1 \quad 6 \quad 4}_{ms}$

$\underbrace{5 \quad 2}_{merge} \qquad 1 \qquad \underbrace{6 \quad 4}_{ms}$

$[2 \quad 5]$

$\underbrace{6 \quad 4}_{merge}$

$\underbrace{\quad 4 \quad 6}_{merge}$

$[1 \quad 4 \quad 6]$

$\underbrace{\qquad\qquad\qquad\qquad}$

$1 \quad 2 \quad 4 \quad 5 \quad 6$

Quick Sort

- Select Pivot

  Start at left and right ends of array

  - move L until $A[L] \geq A[P]$
  - move R until $A[R] \leq A[P]$
  - Swap value

  insert Pivot

  repeat

75	5	55	88	78	42	18	93
↑			↑			↑	
P			L			R	

75	5	55	18	78	42	88	93
↑				↑	↑		
P				L	R		

75	5	55	18	42	78	88	93
↑				↑	↑		
P				L	R		

42 | 5 | 55 | 18 | |75| | 78 | 88 | 93

L P R

QS QS

42 5 55 18 |75| 78 88 93

P L R P LR

42 5 18 55 |75| |78| 88 93

P L R P QS

18 5 |42| 55 |75| |78| |88| |93|

P L R QS QS

|5| |18| |42| |55| |75| |78| |88| |93|

$O(n \log(n))$

$1 \quad \underbrace{2 \quad 3 \quad 4}_{QS} \quad 5$

p R

$\boxed{1} \quad 2 \quad 3 \quad 4 \quad 5$

p L R

$\boxed{1} \quad \boxed{2} \quad 3 \quad 4 \quad 5$

p L R

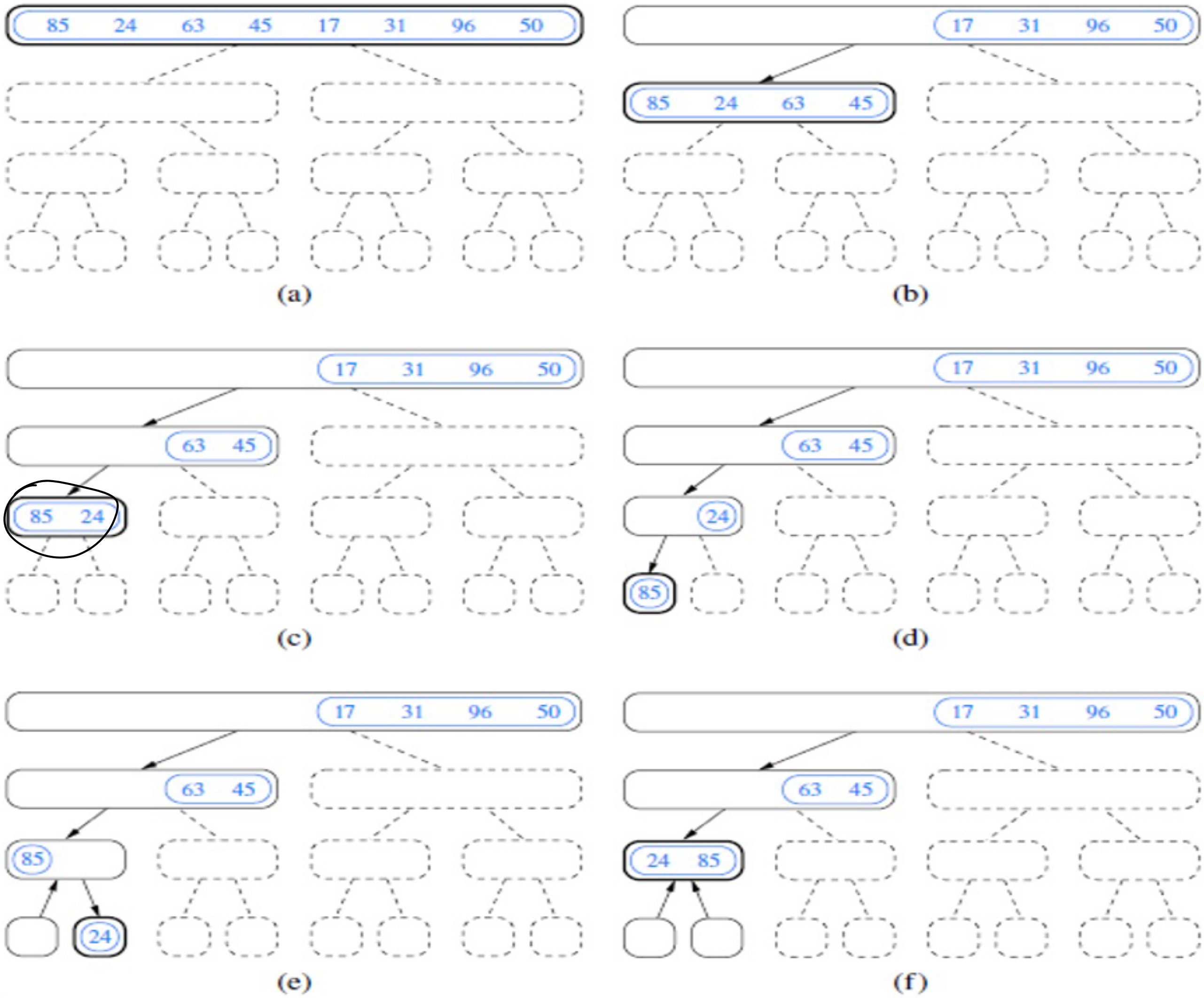
$\boxed{1} \quad \boxed{2} \quad \boxed{3} \quad 4 \quad 5$

p LR

$\boxed{1} \quad \boxed{2} \quad \boxed{3} \quad \boxed{4} \quad \boxed{5}$

p LR

$O(n^2)$


85 24 63 45 17 31 96 50
(a)
17 31 96 50
85 24 63 45
(b)
17 31 96 50
63 45
85 24
(c)
17 31 96 50
63 45
24
85
(d)
17 31 96 50
63 45
85
24
(e)
17 31 96 50
63 45
24 85
(f)

Suppose we are given two n-element sorted sequences A and B that may contain duplicate entries. Describe an O(n) time method for computing a sequence representing the set A∪B with no duplicates

1 2 4 9 15

1 9 12 15 26

1 2 4 9 12 26

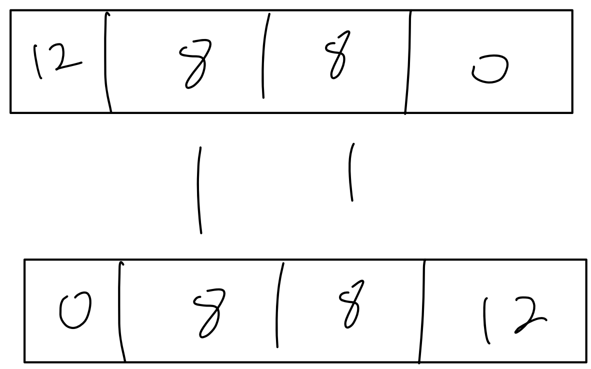
Describe the kind of sequence that would cause the quick-sort algorithm to run in $\Omega(n^2)$.

A sorted array assuming pivot is first element

•

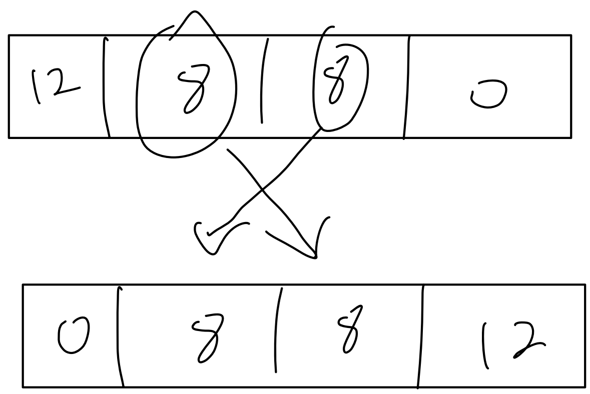
Basis for comparison	Quick sort	Merge sort
The partition of elements in the array	watch the	PL
Additional storage space requirement	Less (in place)	more (not in place)
Efficiency	bad for large	good for large
Sorting method	internal	external
Stability	Not stable	Stable
Preferred for	array	linked list
Locality of reference	good	Poor

stablity



constants are not changed

not stable



constants may be changed

	Quick-Sort	**Merge Sort**
Worst-Case Performance	$O(n^2)$	O(n log n)
Average-Case Performance	O(n log n)	O(n log n)
Best-Case Performance	O(n log n)	O(n log n)
In-Place	Yes	Not Traditionally
Stable	Not Traditionally	Yes